[MKT212]

Numerical Methods in Mechatronics Engineering 2nd Homework

05/06/2020

180223028 - Buğra Coşkun

bgra.coskn@gmail.com

Regression

x_i	0	2	4	8	10	12	20
y_i	5	6	10	13	12	8	6

Table 1

[1] Fit a straight line to the data given in Table I. Compute the equation and r^2 coefficient determination and plot the computed line with the data in Table I. Comment on the results.

Using the Least Squares Linear Regression method lets say that we have a line with the equation of $y_p = a_0 + a_1 x$ The y_p stands for predicted y value. I will most likely use this convention throughout the homework.

We should find the values $a_0 \; and \; a_1$ that minimizes the value:

$$S_r = \sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n} \left(y_i - y_p\right)^2$$

Notice that I use y_p to indicate both the predicted line equation and the scalar value it returns when inserted with an x value.

Thanks to being linear we don't need to pull out any fancy algorithms like gradient-descent. We can use the formulas already given to us, albeit lazily.

$$a_1 = \frac{n\sum x_i y_i - \sum x_i \sum y_i}{n\sum x_i^2 - (\sum x_i)^2} \quad and \quad a_0 = \bar{y} - a_1 \bar{x}$$

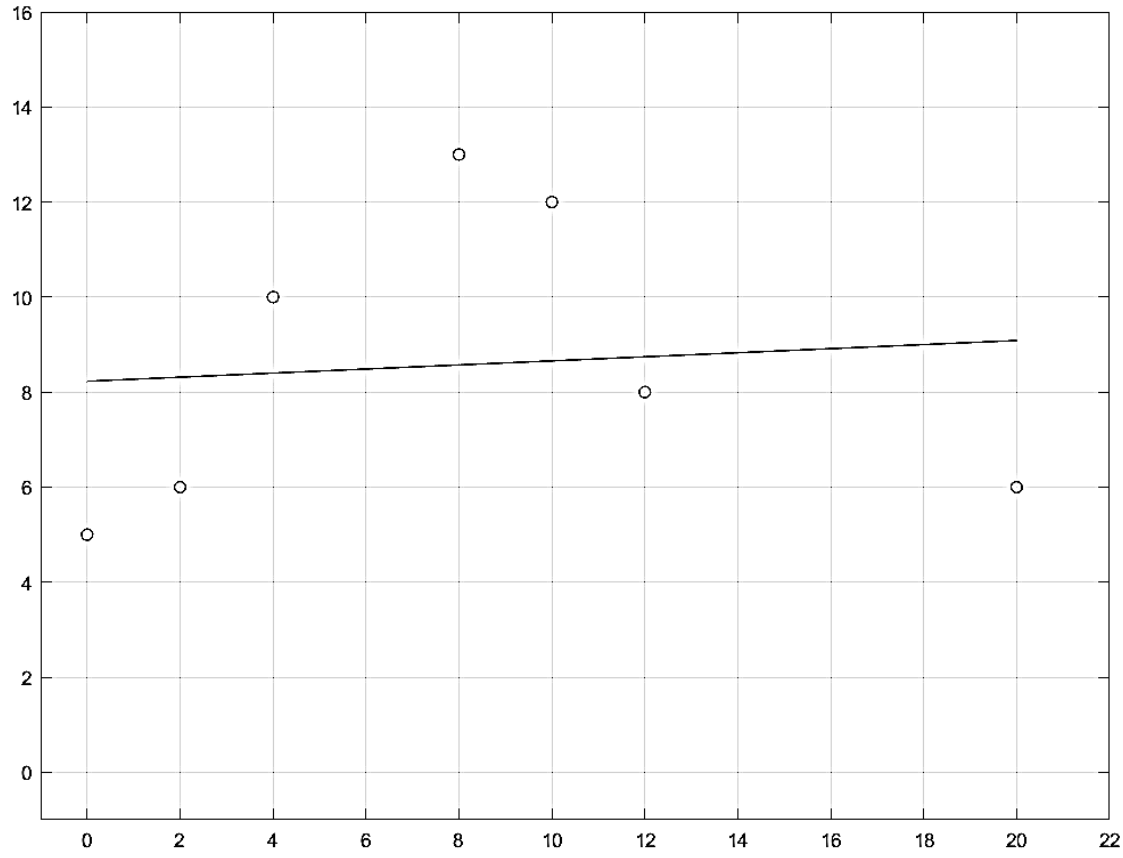
We know that: $n = 7 \quad \bar{y} = 8.5714 \quad \bar{x} = 8$ therefore

$$a_1 = \frac{(7 \cdot 492 - 3360)}{7 \cdot 728 - 3136} \quad \therefore \quad \boldsymbol{a_1 \approx 0.0429}$$

$$a_0 = 8.5714 - 0.0429 \cdot 8 \quad \therefore \quad \boldsymbol{a_0 \approx 8.2286}$$

Just to make sure I also used the *polyfit* function of matlab to check the values and yes the values are spot on. In conclusion the linear regression gave us a line prediction equation:

$$\boldsymbol{y_p = 8.2286 + 0.0429x}$$



(Data and Calculated Line)

To represent how well our calculated line fits the data we should find the r^2 coefficient:

$$r^2 = \frac{S_t - S_r}{S_t} \qquad S_t = \sum (y_i - \bar{y})^2 \qquad S_r = \sum_{i=1}^{n} (y_i - y_p)^2$$

Where S_r is the "sum of the squares of residuals around the regression line" and S_t is the "total sum of the squares around the mean". So we can compare how well our fitted line works against a straight line with the mean of our data as a constant *y* value.

Calculating S_t and S_r using a matlab loop for convenience sake:

```
mean_Y = mean(Y);
ST = 0; SR = 0;
for t = 1:length(Y)
    ST = ST + (Y(t) - mean_Y)^2;
    SR = SR + (Y(t) - a_0 - a_1*X(t))^2;
end
```

$$\therefore \quad \boxed{S_t = 59.7143 \quad \& \quad S_r = 59.2}$$

Thus our r^2 correlation coefficient is:

$$r^2 = \frac{59.7143 - 59.2}{59.7143} \quad \therefore \quad \boldsymbol{r^2 = 8.61 \times 10^{-3}}$$

Our r^2 coefficient is pretty close to 0, meaning our linear regression doesn't really improve on just taking the mean of the data for the line equation. But that makes sense, if we visually analyze the data we can see that the distribution isn't linear at all.

Also remember that the mean of the Y array was actually 8.5714 and our regression line is pretty close to it as well, no wonder it didn't really improve our predictions.

$$S_t \rightarrow y_p = 8.2286 + 0.0429x \qquad \& \qquad S_r \rightarrow y_m = 8.5714$$

We can see that the sum errors of their respected line equations are close in value because the line equations themselves are really similar.

So, it looks like a higher order polynomial regression would fit our dataset a lot better. But of course if the regression polynomial has a too high of an order it could overfit.

[2] Fit a second order polynomial to the data given in Table I. Calculate the least-squares quadratic equation and coefficient determination (r2) and plot the quadratic equation with the data in Table I.

Very similarly to the first question lets say we have a prediction function:

$$y_p = a_0 + a_1 x + a_2 x^2$$

We should find the values a_0 , a_1 and a_2 that minimizes the function:

$$S_r = \sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n} (y_i - a_0 - a_1 x_i - a_2 x_i^2)^2$$

We can calculate the values below, to then put them in an appropriate matrix that will give us a system of linear equations which we can calculate with Gauss elimination method to find the a_0 , a_1 and a_2 values.

$$n = 7 \qquad \sum x_i = 56 \qquad \sum x_i^4 = 195104$$

$$m = 2 \qquad \sum y_i = 60 \qquad \sum x_i y_i = 492$$

$$\bar{x} = 8 \qquad \sum x_i^2 = 728 \qquad \sum x_i^2 y_i = 5768$$

$$\bar{y} = 8.5714 \qquad \sum x_i^3 = 11312$$

So we can write the system of linear equations as[1]:

$$\begin{bmatrix} 7 & 56 & 728 \\ 56 & 728 & 11312 \\ 728 & 11312 & 195104 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} 60 \\ 492 \\ 5768 \end{bmatrix}$$

Using Gauss elimination method to transform the augmented matrix into a reduced row echelon form:

$$\begin{bmatrix} 67 & 56 & 728 & 60 \\ 56 & 728 & 11312 & 492 \\ 728 & 11312 & 195194 & 2892 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 0 & 0 & 5.0714 \\ 0 & 1 & 0 & 1.2148 \\ 0 & 0 & 1 & -0.0598 \end{bmatrix}$$

[1] *Chapra, S. C. and Raymond, P. C. Numerical Methods for Engineers.7th ed., pp.473 (17.19)*

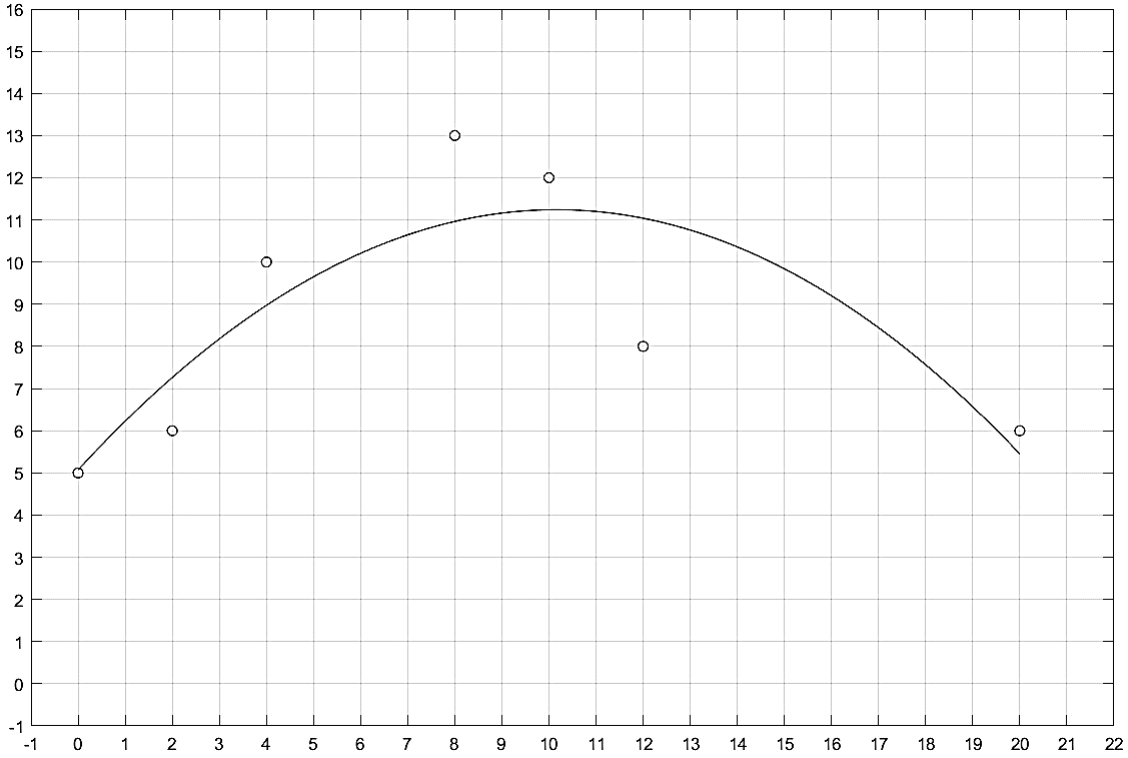
Solving with Gauss Elimination Method we find the values a_0 , a_1 and a_2 as:

$$a_0 = 5.0714 \quad a_1 = 1.2148 \quad a_2 = -0.0598$$

Meaning our polynomial regression function (prediction function) will be:

$$\boxed{\boldsymbol{y_p(x) = 5.0714 + 1.2148x - 0.0598x^2}}$$

And when we draw it on the plot with the original data points:



We can see it's a proper estimation for a second order polynomial. If the last data point at x = 20 didn't exist (because it deviates so much from the pattern of data points before) a second order polynomial regression would fit the data very well (drawing a concave parabola with a global max around x = 8). But because of the last data point, a spline interpolation or a higher order polynomial could have been more suitable for a better fit to our data.

For the r^2 (correlation) coefficient:

$$r^2 = \frac{(y_i - \bar{y})^2 - \left(y_i - y_p\right)^2}{(y_i - \bar{y})^2} = \frac{46.2857 - 2.1158}{46.2857} \therefore \quad \boxed{\boldsymbol{r^2 = 0.9543}}$$

Thus it seems our second order polynomial explains 95.43% of the dataset's uncertainty

Interpolation

x_i	4	8	10	12	20
y_i	10	13	12	8	6

Table II

[3] Use 1st and 2nd order Lagrange interpolations to evaluate x = 9 on the basis of the data given in Table II.

As it can be seen from the table of data, the interpolation points we must use are those with x values 8 and 10, because the point we are evaluating stands in between.

The values we will need:

$$p_0 = 8 \quad p_1 = 10 \quad p_2 = 12 \;\rightarrow\; f(p_0) = 13 \quad f(p_1) = 12 \quad f(p_2) = 8$$

In order to not confuse the points with the data in the table I renamed them to "p".

The first order Lagrange Interpolating Polynomial when evaluating a point at x = 9 can be calculated by:

$$f_1(x) = \frac{x - p_1}{p_0 - p_1} f(p_0) + \frac{x - p_0}{p_1 - p_0} f(p_1)$$

$$\rightarrow f_1(9) = \frac{9 - 10}{8 - 10} \cdot 13 + \frac{9 - 8}{10 - 8} \cdot 12 \quad \therefore \quad \boxed{\boldsymbol{f_1(9) = 12.5}}$$

And the second order Lagrange Interpolating Polynomial is:

$$f_2 = \frac{(x - p_1)(x - p_2)}{(p_0 - p_1)(p_0 - p_2)} f(p_0) + \frac{(x - p_0)(x - p_2)}{(p_1 - p_0)(p_1 - p_2)} f(p_1) + \frac{(x - p_0)(x - p_1)}{(p_2 - p_0)(p_2 - p_1)} f(p_2)$$

Quite the minimalist formula for sure (!) Omitted the x on $f_2(x)$ because the display would look funky otherwise.

For this one I'm going to skip the simple but quite the handful arithmetic to spare some time and space:

$$\rightarrow \boxed{\boldsymbol{f_2(9) = 12.875}}$$

x_i	4	8	10	12	20
y_i	10	13	12	8	6

Copy, Table II

[4] Fit quadratic splines to the data given in Table II. Plot the quadratic splines together with data in Table II.

For this problem I mainly relied on the chapter Spline Interpolations in Chapra's book for it's more detailed look into the logic of the method instead of the more simplified and practical formulations of the lecture slides[2]*. Though I did tweak some of the details to better fit the problem.*

As it stands we have 5 points that are given, meaning 4 intervals, 12 coefficients and 12 equations we must solve using the following:

This will give us the polynomials for each interval:

$$a_i x_i^2 + b_i x_i + c_i = f(x_i) \qquad i = 0,1,\dots,n$$

The steps we must follow will be:

1. For the internal knots the function values must be equal for adjacent polynomials, thus:

$$a_{i-1}x_{i-1}^2 + b_{i-1}x_{i-1} + c_{i-1} = f(x_{i-1}) \quad and$$
$$a_i x_{i-1}^2 + b_i x_{i-1} + c_i = f(x_{i-1}) \quad for \quad i = 2,3,\dots,n$$

2. The first and last polynomials should pass through the first and last data point as well, so:

$$a_1 x_0^2 + b_1 x_0 + c_1 = f(x_0) \quad and \quad a_n x_n^2 + b_n x_n + c_n = f(x_n)$$

3. For adjacent polynomials at the interior nodes the first derivatives should be equal:

$$2a_{i-1}x_{i-1}^2 + b_{i-1} = 2a_i x_{i-1} + b_i \qquad i = 2,\dots,n$$

4. Lastly we can make an arbitrary decision that the second derivative is 0 at the first data point: $a_1 = 0$

After using all these equations we can form a linear system of equations which we can solve using the Gauss Elimination Method like before.

$$n = 4 \quad \because \quad \#points = n + 1 = 5$$

$$x_0 = 4 \quad x_1 = 8 \quad x_2 = 10 \quad x_3 = 12 \quad x_4 = 20$$

$$f(x_0) = 10 \quad f(x_1) = 13 \quad f(x_2) = 12 \quad f(x_3) = 8 \quad f(x_4) = 6$$

[2] *Chapra, S. C. and Raymond, P. C. Numerical Methods for Engineers.7th ed., pp.511-517*

First step, equality of adjacent polynomials at internal knots:

$$i = 2 \quad \rightarrow \quad 64a_1 + 8b_1 + c_1 = 64a_2 + 8b_2 + c_2 = 13$$

$$i = 3 \quad \rightarrow \quad 100a_2 + 10b_2 + c_2 = 100a_3 + 10b_3 + c_3 = 12$$

$$i = 4 \quad \rightarrow \quad 144a_3 + 12b_3 + c_3 = 144a_4 + 12b_4 + c_4 = 8$$

Second step, the first and last polynomials should pass through the first and last data points respectively:

$$16a_1 + 4b_1 + c_1 = 10 \quad and \quad 400a_4 + 20b_4 + c_4 = 6$$

Third step, similar to the first step, this time the first derivatives at the internal knots should be equal:

$$i = 2 \quad \rightarrow \quad 16a_1 + b_1 = 16a_2 + b_2$$

$$i = 3 \quad \rightarrow \quad 20a_2 + b_2 = 20a_3 + b_3$$

$$i = 4 \quad \rightarrow \quad 24a_3 + b_3 = 24a_4 + b_4$$

Fourth step, $a_1 = 0$

With that we have all the 12 equations we should solve for to find the coefficients.

The reduced echelon form would be the augmented matrix $(\, I_{12} \mid S \,)$ where I_{12} is the identity matrix of 12×12 dimensions and S is the column vector of the values of coefficients with dimension 12×1. Using the convenient command of reshape in matlab lets turn the array into a neat matrix of coefficient values.

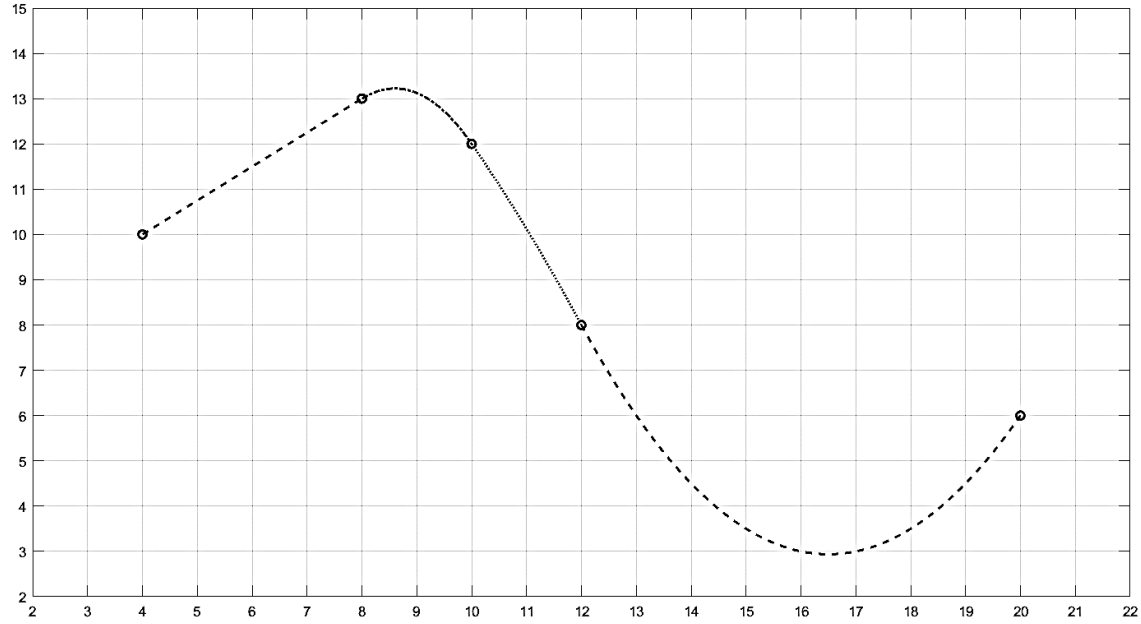
```
>> S_m = reshape(S,[3,4]);
```

$$\begin{bmatrix} a_1 & a_2 & a_3 & a_4 \\ b_1 & b_2 & b_3 & b_4 \\ c_1 & c_2 & c_3 & c_4 \end{bmatrix} = \begin{bmatrix} 0 & -5/8 & -1/8 & 1/4 \\ 3/4 & 43/4 & 3/4 & -33/4 \\ 7 & -33 & 17 & 71 \end{bmatrix}$$

So our quadratic splines will be:

$$\boldsymbol{f_1(x) = 0.75x + 7}$$
$$\boldsymbol{f_2(x) = -0.625x^2 + 10.75x - 33}$$
$$\boldsymbol{f_3(x) = -0.125x^2 + 0.75x - 17}$$
$$\boldsymbol{f_4(x) = 0.25x^2 - 8.25 + 71}$$

And if we plot the splines with the original data:



Where, as it can be seen the black circles represent the data points and the different types of lines represent the polynomials for each interval.

And that's how the total spline fit looks. There are some problems mentioned when it comes to quadratic splines such as how the first polynomial is just a line when a concave curve could have fit the dataset better or how the last polynomial tends to swing more than what might have been necessary and such. When it comes to such problems cubic splines are said to be more fitting.

Numerical Integral

$$f(x) = 1.73x^4 - 9x^3 + 5.7x^2 + 0.5$$

[5] Use the f(x) and integrate 0 to 0.8 by

a) Single application of the trapezoidal rule
b) 8-segment trapezoidal rule
c) Simpson's 1/3 rule
d) Simpson's 3/8 rule
e) Compare the results with the analytical solution and comment on the results.

[5.a] The trapezoidal rule is a method where we just draw a straight line between the boundaries in which we integrate the function and calculate the area under it, so basically the area of the trapezoid:

$$I = \int_a^b f(x)dx \cong \int_a^b f_1(x)dx$$

The values we will need to use for this problem are:

$$a = 0 \quad b = 0.8 \quad f(a) = 0.5 \quad f(b) = 0.2486$$

And we can represent the straight line forming the trapezoid as $f_1(x)$ to find the area under it we get it's integral, but we don't need to do that as part of the trapezoidal rule the integrated formula is already given to us:

$$f_1(x) = f(a) + \frac{f(b) + f(a)}{b - a}(x - a) \quad \rightarrow \quad I = (b - a)\frac{f(a) + f(b)}{2}$$

$$I = (0.8 - 0)\frac{0.5 + 0.8}{2} = 0.52 \quad \therefore \quad \boxed{I \cong 0.52}$$

[5.b] Using the multiple-application trapezoidal rule with 8 segments, we first divide the area that we integrate into $h = 0.1$ length segments. Also we need $n + 1 = 9$ equally spaced points for the segments.

$$x_0 = 0 \quad x_1 = 0.1 \quad \ldots \quad x_8 = 0.8$$

$$I \cong \frac{h}{2}\left[f(x_0) + 2\sum_{i=1}^{n-1} f(x_i) + f(x_n)\right] = 0.05 \cdot (0.5 + 2 \cdot 5.2329 + 0.2486) = 0.5607$$

$$\boxed{I \cong 0.5607}$$

[5.c] Simpson's 1/3 rule is a method where we take three points and use a second order polynomial to approximate the curve that we are trying to approximately integrate and take the area under for our approximate integration result. Instead of trying to recreate the rule, if we use the simplified formula:

$$I \cong \int_a^b f_2(x)dx \;\rightarrow\; I \cong (b-a)\frac{f(x_0)+4f(x_1)+f(x_2)}{6} \quad where \quad \begin{matrix} x_0 = 0 \\ x_1 = 0.4 \\ x_2 = 0.8 \end{matrix}$$

$$I \cong\ 0.8 \cdot \frac{0.5 + 3.5212 + 0.2486}{6} \rightarrow \boxed{I \cong 0.5693}$$

[5.d] Simpson's 3/8 rule is very similar to 1/3 rule, but instead of a second order polynomial it uses a cubic equation connecting 4 points.

$$I \cong (b-a)\frac{f(x_0)+3f(x_1)+3f(x_2)+f(x_3)}{8} \quad where \quad \begin{matrix} x_0 = 0 \\ x_1 = 0.2667 \\ x_2 = 0.5333 \\ x_3 = 0.8 \end{matrix}$$

$$I \cong 0.8 \cdot \frac{0.5 + 3 \cdot 0.7435 + 3 \cdot 0.8960 + 0.2486}{8} \rightarrow \boxed{I \cong 0.5667}$$

[5.e] Finally lets compare the approximate results of the integration with the analytical result by finding the relative error percentage:

$$f(x) =\ 1.73x^4 - 9x^3 + 5.7x^2 + 0.5 \;\rightarrow \int_a^b f(x)dx\ = 0.5646$$

Single Application Trapezoid Rule:

$$\varepsilon_t = \frac{|0.5646 - 0.52|}{0.5646} \times 100\% = 7.9\%$$

Multiple Application Trapezoid Rule:

$$\varepsilon_t = \frac{|0.5646 - 0.5607|}{0.5646} \times 100\% = 0.69\%$$

Simpson's 1/3 Rule:

$$\varepsilon_t = \frac{|0.5646 - 0.5693|}{0.5646} \times 100\% = 0.83\%$$

Simpson's 3/8 Rule:

$$\varepsilon_t = \frac{|0.5646 - 0.5667|}{0.5646} \times 100\% = 0.37\%$$

As shown, Simpson's 3/8 rule had the lowest relative error and just behind it is the multiple application trapezoid rule. Overall due to their nature, the results weren't very surprising.

Numerical Differentiation

$$f(x) = 1.73x^4 - 9x^3 + 5.7x^2 + 0.5$$

[6] Calculate the derivative of f(x) at x = 0.3 using finite divided differences and a step size of h = 0.1

a) Forward difference of accuracy of O(h) and O(h^2)
b) Backward difference of accuracy of O(h) and
c) Centered difference of accuracy of O(h) and
d) Richardson Exrapolation
e) Compare the results with the analytical solution

[6.a] Forward difference, first derivatives:

$$x_i = 0.3 \quad x_{i+1} = 0.4 \quad x_{i+2} = 0.5$$

$$f(x_i) = 0.784 \quad f(x_{i+1}) = 0.8803 \quad f(x_{i+2}) = 0.9081$$

$$f'(x_i) = \frac{f(x_{i+1}) - f(x_i)}{h} + O(h) \quad f'(x_i) = \frac{-f(x_{i+2}) + 4f(x_{i+1}) - 3f(x_i)}{2h} + O(h^2)$$

For forward difference of accuracy $O(h)$:

$$f'(0.3) = \frac{0.8803 - 0.784}{0.1} = \mathbf{0.963} \quad \boxed{\varepsilon_t = 18.17\%}$$

For forward difference of accuracy $O(h^2)$:

$$f'(x) = \frac{-0.9081 + 4 \cdot 0.8803 - 3 \cdot 0.784}{0.2} = \mathbf{1.3055} \quad \boxed{\varepsilon_t = 10.94\%}$$

[6.b] Backward difference, first derivatives:

$$x_i = 0.3 \quad x_{i-1} = 0.2 \quad x_{i-2} = 0.1$$

$$f(x_i) = 0.784 \quad f(x_{i-1}) = 0.6588 \quad f(x_{i-2}) = 0.5482$$

$$f'(x_i) = \frac{f(x_i) - f(x_{i-1})}{h} + O(h) \quad f'(x_i) = \frac{3f(x_i) - 4f(x_{i-1}) + f(x_{i-2})}{2h} + O(h^2)$$

For backward difference of accuracy $O(h)$:

$$f'(0.3) = \frac{0.783 - 0.6588}{0.1} = \mathbf{1.242} \quad \boxed{\varepsilon_t = 5.54\%}$$

For backward difference of accuracy $O(h^2)$:

$$f'(0.3) = \frac{3 \cdot 0.784 - 4 \cdot 0.6588 + 0.5482}{0.2} = \mathbf{1.325} \quad \boxed{\varepsilon_t = 12.59\%}$$

[6.c] Centered difference, first derivatives:

$$x_i = 0.3 \quad x_{i-1} = 0.2 \quad x_{i-2} = 0.1 \quad x_{i+1} = 0.4 \quad x_{i+2} = 0.5$$

$$f(x_i) = 0.784 \quad f(x_{i-1}) = 0.6588 \quad f(x_{i-2}) = 0.5482$$

$$f(x_{i+1}) = 0.8803 \quad f(x_{i+2}) = 0.9081$$

$$f'(x_i) = \frac{f(x_{i+1}) - f(x_{i-1})}{2h} + O(h^2)$$

$$f'(x_i) = \frac{-f(x_{i+2}) + 8f(x_{i+1}) - 8f(x_{i-1}) + f(x_{i-2})}{12h} + O(h^4)$$

For centered difference of accuracy $O(h^2)$:

$$f'(0.3) = \frac{0.8803 - 0.6588}{0.2} = \mathbf{1.1075} \quad \boxed{\varepsilon_t = 5.89\%}$$

For centered difference of accuracy $O(h^4)$:

$$f'(0.3) = \frac{-0.9081 + 8 \cdot 0.8803 - 8 \cdot 0.6588 + 0.5482}{1.2} = \mathbf{1.1768} \quad \boxed{\varepsilon_t = 0\%} *$$

**Well that was unexpected, for our precision of 4 significant figures, centered difference of accuracy O(h⁴) results in the same value as the analytical solution.*

[6.d] Richardson Extrapolation, first derivatives:

To use the Richardson Extrapolation we need to compute two first derivatives of the function to improve upon them, usually we would use the centered difference method but for our precision the centered difference method already gives us a relative error of 0% ! So I will re-calculate the centered difference method of accuracy $O(h^4)$ again with more precision:

$$h_1 = 0.1 \ \ and \ \ h_2 = 0.05$$

$$h_1 \rightarrow f(x_{i+1}) = 0.8803 \ \ f(x_{i-1}) = 0.6588$$

$$h_2 \rightarrow f(x_{i+1}) = 0.8383 \ \ f(x_{i-1}) = 0.7224$$

$$f'(0.3) \cong D(h_1) = \frac{0.8803 - 0.6588}{0.2} = \mathbf{1.1075} \quad \boxed{\varepsilon_t = 5.89\%}$$

$$f'(0.3) \cong D(h_2) = \frac{0.8383 - 0.7224}{0.1} = \mathbf{1.159} \quad \boxed{\varepsilon_t = 1.51\%}$$

So the improved estimate will be:

$$D = \frac{4}{3}(1.159) - \frac{1}{3}(1.1075) = \mathbf{1.1762} \quad \boxed{\varepsilon_t = 0.05\%}$$

Using two centered difference approximations with $O(h^2)$ accuracy we got a relative error of only 0.05%

[6.e] Comparing the approximation results with the analytical solution:

Using analytical methods we find that $f'(0.3) = 1.1768$ but because that I **already calculated the relative errors for each method with their approximate result** I will only comment on the outcomes instead.

Interestingly backward difference of accuracy $O(h)$ had a better approximation than the forward difference of accuracy $O(h^2)$. Overall the best result was centered difference of accuracy $O(h^4)$ with a whopping 0% relative error for 4 significant figure precision. Even better than the Richardson Extrapolation method's result.